The Inside Story behind Interactive Supercomputing's Star-P Platform for High Performance Computing for MATLAB(r)

Alan Edelman
Massachusetts Institute of Technology
Professor of Applied Mathematics
Computer Science and AI Laboratory
Interactive Supercomputing
Chief Science Officer
Company

• Background:
  – Started in 1995, Founded in 2004
  – Parallel Computing Harder than most realize
  – Technology: Star-P software platform supporting automatic parallelization and interactive execution of desktop technical applications on parallel servers
  – Platform: Clients: MATLAB, MATHEMATICA, PYTHON
  – Platform: Engines, your code, etc.

• Value:
  – Modern Client/Server Parallel Computation
  – OPEN PLATFORM
  – Can plug in existing parallel and serial software seamlessly
  – Years of experience
Client/Server Parallel Computing
The Client (a math lab) is the browser

Web vs traditional
- Bank/financial
- Email
- Travel
- Photos
- MIT Grades
- Your Parallel Computing!
Star-P Functional Overview

![Star-P Functional Overview Diagram]
Familiar Desktop Tools
Star-P Client

- Connects to server
- Redirects library calls
- Optimizes serial code
Star-P Interactive Engine

- Server resource management
- User & session management
- Workload management
Star-P Computation Engine

1. Data-Parallel Computations
2. Task-Parallel Computations
3. OpenConnect Library API Link
Data-Parallel Computations

- Global array syntax
- Operations on large distributed data sets
- World-class parallel libraries
Brings It All Together!
ppeval syntax (parallel function)

- \( a = \text{rand}(500,500,200*p); \)
- \([u,s,v]=\text{ppeval}('svd',a); \) % default svd on z-dim
- \( a = \text{rand}(500,500*p,200); \)
- \([u,s,v]=\text{ppeval}('svd',a); \) % default svd on z-dim

Answer does not depend on distribution:
Parallel computers need shapes to enter from all sides.
Task-Parallel Computations

- Multiple independent calculations
- Simple, intuitive w/Star-P’s abstraction
- Plug in popular computation engines
Star-P OpenConnect Library API Link

- Leverage data- and task-parallel libraries, solvers
- Commercial and open source
- Enable access through desktop VHLLs
Star-P OpenConnect Library API

- Leverage data- and task-parallel libraries, solvers
- Commercial and open source
- Enable access through desktop VHLLs
Hardware Accelerators

• Embed compute-intensive algorithms
• FPGAs, GPUs, etc.
• Library functions, called from desktop apps
Development Utilities

• Debugging, profiling, monitoring
• Built in, and interfaces to popular tools
• Interactively explore and optimize code
High-speed I/O

- Native parallel I/O
- Direct transfer between disk and server CPUs
- Eliminate client/server data transfer
- No need to manually break up files
Classroom Homework

• The Buffon Needle Problem

Buffon(1,1,1.5,1000*p)

function z=Buffon(a,b,l, trials)
    r=rand(trials,3);
    x=a*r(:,1)+l*cos(2*pi*r(:,3));
    y=b*r(:,2)+l*sin(2*pi*r(:,3));
    inside = (x >= 0) & (y>=0) & (x <= a) & (y <= b);
    buffonpi=(2*l*(a+b) - l^2)/ (a*b*(1-sum(inside)/trials));
Classroom Experiment

• A data collector’s dream:
  – 29 students, each code run in MPI and three versions of Star-P. Some students more skilled with MPI than others.
Classroom Experiment

- A data collector’s dream:
  - 29 students, each code run in MPI and three versions of Star-P. Some students more skilled with MPI than others.

<table>
<thead>
<tr>
<th>Star-P 2.1 (March 2006)</th>
<th>Star-P 2.3 (May 2006)</th>
<th>Star-P internal</th>
</tr>
</thead>
</table>

![Graph showing Mean mpi Time for different versions of Star-P]
Productivity Study – Kepner diagram

- Star-P internal
- Star-P 2.3
- Star-P 2.1

 MPI Best
MPI Typical
The silly (worse than embarrassing) pi example (followed by the good one)

```
>> n=8;
>> sum
```

```
Parallel Evaluate Pieces of pi:
\[
\int \frac{4}{1+x^2} \, dx \text{ on } [0,1/8], [1/8,2/8], \ldots, [7/8,1] \text{ and sum.}
\]
```

```
ans =
3.14159265358979
```

```
function thedigits = pidigits(d)
sum1 =0; sum2 = zeros(4);
A = eye(d+1,d+1); B = zeros(d+1,1); n = 1;
g = [1,4,5,6];
for m = g
    if (m == 1),A(1) =0; end
    for j = 0:d
        B(j+1,1) = 8*j+m;
        for i = j+1:d
            A(i+1,j+1) = mod(A(i, j+1)*16, 8*j+m);
        end
    end
    for i = 1:d+1, f(i,n) = sum(A(i,:)); end
    n = n+1; u = f-floor(f); A = eye(d+1,d+1);
end
for e = 0:d
    for k = d+1:d+20
        b = 16^(d-k)./(8*k+[1 4 5 6]);
        sum1 = sum1 + (b-floor(b));
    end
    sum2(e+1,1:4) = sum1;
end
q = u + sum2; soln = 4*q(:,1)-2*q(:,2)-q(:,3)-q(:,4);
thedigits = floor(16*(soln - floor(soln)));
```

Compute millions of hexadecimal digits of pi!
Wigner’s semicircle Law with four clients

Take Random Symmetric Matrix and histogram the eigenvalues
Famous Noble Prize Winning Physicist
Computed histogram = semicircle
MATLAB

>> n=2000;
>> a=randn(n*p); s=(a+a')/(sqrt(8*n)); e=eig(s,'sym');
>> [y,x]=hist(ppfront(e),25); bar(x, (y/n)/(x(2)-x(1)))
>> x=-1:.01:1; hold on; plot(x,(2/pi)*sqrt(1-x.^2),'r','LineWidth',5)
<< Statistics`NormalDistribution`
<< Graphics`Graphics`
n = 2000;
a = RandomArray[NormalDistribution[], {n, n × P}];
s = (a + Transpose[a]) / Sqrt[s × n];
e = Eigenvalues[s];

In[56]:= hist = Histogram[e, HistogramCategories → 25, HistogramScale → 1];
semicircle = Plot[(2 / Pi) × Sqrt[1 - x^2], {x, -1, 1}, DefaultColor → Red];
Show[hist, semicircle]
Python

Python 2.5 (r25:51908, Sep 19 2006, 09:52:17) [MSC v.1310 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.

******************************************************************************
Personal firewall software may warn about the connection IDLE
makes to its subprocess using this computer's internal loopback
interface. This connection is not visible on any external
interface and no data is sent to or received from the Internet.
******************************************************************************

IDLE 1.2
>>> from numpy import *; from pylab import *; from matplotlib import *
>>> n=2000;
>>> a=randn(n,n)+s=(a+transpose(a))/sqrt(8*pi);e=linalg.eigvalsh(s);
>>> hist(e,25,normed=1);
>>> x=linspace(-1,1,201);y=(2/pi)*sqrt(1-x*x);
>>> plot(x,y,'r',linewidth=3);

Figure 3
R Client

R version 2.4.0 (2006-10-03)
Copyright (C) 2006 The R Foundation
ISBN 3-900051-07-0

R is free software and comes with ABSOLUTELY NO
warranty. You are welcome to redistribute it
Type 'license()' or 'licence()' for
Natural language support but runner

R is a collaborative project with many
Type 'contributors()' for more info
'citation()' on how to cite R or R packages
Type 'demo()' for some demos, 'help
'help.start()' for an HTML browser
Type 'q()' to quit R.

> n<-2000;
> a<-matrix(rnorm(n*n),nrow=n,ncol=n);s<-(-t(a))/sqrt(8*n);
> e=eigen(s,symmetric=T,only.values=T)$values;
> hist(e,25,freq=F,col='blue');curve((2/pi)*sqrt(1-x^2),-1,1,col='red',lwd=5,add=T)
Star-P Functional Overview